

From Tweets to Events: Exploring a Scalable Solution for Twitter Streams

Shamanth Kumar[†], Huan Liu[†], Sameep Mehta^{*}, and L. Venkata Subramaniam^{*}

[†] Computer Science & Engineering, CIDSE, Arizona State University, Tempe, AZ

^{*} IBM India Research Lab, New Delhi, India

{shamanth.kumar, huan.liu}@asu.edu, {sameepmehta, lvsubram}@in.ibm.com

Abstract

The unprecedented use of social media through smartphones and other web-enabled mobile devices has enabled the rapid adoption of platforms like Twitter. Event detection has found many applications on the web, including breaking news identification and summarization. The recent increase in the usage of Twitter during crises has attracted researchers to focus on detecting events in tweets. However, current solutions have focused on static Twitter data. The necessity to detect events in a streaming environment during fast paced events such as a crisis presents new opportunities and challenges. In this paper, we investigate event detection in the context of real-time Twitter streams as observed in real-world crises. We highlight the key challenges in this problem: the informal nature of text, and the high volume and high velocity characteristics of Twitter streams. We present a novel approach to address these challenges using single-pass clustering and the compression distance to efficiently detect events in Twitter streams. Through experiments on large Twitter datasets, we demonstrate that the proposed framework is able to detect events in near real-time and can scale to large and noisy Twitter streams.

Introduction

Social networking sites like Twitter have proven to be popular outlets for information dissemination during crises. It has been observed that information related to a crisis is released on social media sites before traditional news sites (Gilgoff and Lee 2013), (Khamadi Were 2013). During the Arab Spring movement, Twitter was used as an information source to coordinate protests and to bring awareness to the atrocities (Huang 2011). In recent world events, social media data has been shown to be effective in detecting earthquakes (Sakaki, Okazaki, and Matsuo 2010), rumors (Mendoza, Poblete, and Castillo 2010), and identifying characteristics of information propagation (Qu et al. 2011). This motivates us to study the problem of event detection, which is an interesting and important problem in this domain.

Event detection approaches designed for documents cannot be directly applied to tweets due to the difference in the characteristics of tweets. Unlike a traditional document stream, a Twitter stream suffers from the informality of language, and differs in both volume and velocity characteristics. Existing approaches to event detection in tweets focus on the problem in an offline setting, where the corpus is

static and multiple passes can be employed in the solution. However, event detection in streaming environment presents unique challenges, which prevent the direct application of existing approaches. Detecting events in streaming Twitter data has the following new challenges:

Informal use of language: Twitter users produce and consume information in a very informal manner compared with traditional media (Paris, Thomas, and Wan 2012). Misspellings, abbreviations, contractions, and slang are rampant in tweets, which is exacerbated by the length restriction (a tweet can have no more than 140 characters).

Noise: While traditional event detection approaches assume that all documents are relevant, Twitter data typically contains a vast amount of noise and not every tweet is related to an event (Analytics 2009).

Dynamicity: Twitter streams are highly dynamic with high volume and high velocity as typical characteristics. Approximately 400 million tweets are now posted on Twitter every day (Tsukayama 2013). Event detection methods need to be scalable to handle this high volume of tweets.

Social media such as Twitter empower their users to publish information as soon as a real-world event occurs. However, this information is not curated as in the case of traditional documents, such as news article. Whereas, each news article is part of an event, not every tweet is expected to be part of an event, as there is a significant amount of noise and inter-personal communication. In this paper, we address the above challenges through a novel approach which can:

1. Effectively handle the informality of language in a Twitter stream through the selection of an appropriate distance measure;
2. Efficiently detect events without the need for a preset number of events; and
3. Scale to high volume streaming Twitter data.

Problem Definition

Given an ordered stream of tweets $T = t_1, t_2, t_3, \dots$, where each t_i is associated with a timestamp indicating its publication time, we need to detect events $E = e_1, e_2, \dots, e_m$, where $e_j = t_1, t_2, \dots, t_k$, where $t_k \in T$ and $j \in [1, m]$.

Event: An event is formally defined as a set of similar tweets $E = t_1, t_2, \dots, t_k$ with high user diversity.

User Diversity: of an event is the diversity of the user population who contribute to the event. The intuition here is that a diverse user population lends credibility to the event and helps us filter out noise. Entropy is a measure commonly used to compute the amount of information in a text and here we reformulate it to measure user diversity of an event. Given an event e , its *User Diversity* $H(e)$ is

$$H(e) = - \sum_i \frac{n_{u_i}}{n} \log \frac{n_{u_i}}{n}, \quad (1)$$

where u_i is the i th user in the cluster. Here, n_{u_i} is the number of tweets published by user u_i , which are part of the cluster C , and n is the total number of tweets in the cluster.

Hardness of Event Detection

To detect events E , we aim to find a clustering C of the tweets T such that the user diversity of the resulting clusters and the distance between tweets in different clusters is maximized. Let us assume, that the number of events m is known. Then, we can define the objective function as

$$\arg \max_{e \in E} \sum_j (D(e, e_j) + H(e)), \quad s.t. |E| = m \quad (2)$$

where $D(e_i, e_j)$ is the maximum distance between any pair of elements in clusters e_i and e_j computed as

$$D(e_i, e_j) = \max_{a,b} (D(e_i^{(a)}, e_j^{(b)})), \quad s.t. |e_i| = a, |e_j| = b. \quad (3)$$

To show that this function is hard, we prove that it is submodular using the following properties:

Monotone The function is monotone as at each step a tweet is added to the nearest cluster, hence the summation of the distances between clusters cannot decrease.

Diminishing Returns: Let us assume that S and T are two clusters, where $S \subseteq T$. If a tweet x is added to S and T , then the change in $D(S, P)$ and $D(T, P)$, where P is any other cluster follows one of two scenarios:

- If both T and S contain the tweet which maximizes $D(\cdot)$ and x increases the distance to P . Then $D(T \cup x, P) - D(T, P) = D(S \cup x, P) - D(S, P)$,
- Otherwise, if there is a tweet $x' \notin S$ but $x' \in T$, such that $D(S, P) < D(T, P)$, then $D(T \cup x, P) - D(T, P) \leq D(S \cup x, P) - D(S, P)$ because $D(S, P) < D(T, P)$, therefore the gain in the distance should be larger when the new tweet is added to S .

Therefore the function $D(\cdot)$ is submodular as it satisfies both properties. We also know that the Entropy function $H(\cdot)$ is submodular and the summation of submodular functions is submodular (Krause and Golovin 2012). Therefore, the objective function in Equation 2 is submodular. Maximizing a submodular objective function under the cardinality constraint is NP-hard (Krause and Golovin 2012). Therefore, event detection in streams where m is unknown and cannot be determined in a timely fashion, is at least as hard. Later, we will describe our approach which approximates the objective function by assigning tweets to the most similar cluster and determines events using the cluster's user diversity.

Identifying Events

During a real-world event, people use Twitter to tweet and retweet their experiences. Therefore, the information from these users can be aggregated/clustered to detect events. For streaming Twitter data, however, extra care has to be taken because (1) streaming tweets arrive continuously, traditional multi-pass clustering cannot handle streaming data, and (2) the informal language of tweets defies the standard preprocessing of text corpora such as stemming and vectorization.

To handle high volume and high velocity streaming data, clustering approaches must return the clusters in a single pass. Therefore, we require a clustering approach, which does not require multiple passes over data and which can continuously process the tweets as they arrive. In this paper, we use the single-pass clustering technique described in (Rijsbergen 1979), to group related tweets into clusters as they arrive. This incremental clustering approach continually processes tweets as follows:

1. A tweet is compared with all the candidate clusters.
2. The tweet is added to the closest cluster, if the distance to the cluster is below a threshold.
3. Otherwise, a new cluster is created and the tweet is added to it as its first member.

To cluster the tweets, we must choose a distance measure appropriate for the characteristics of a tweet stream. We require that the distance measure: be scalable to high-volume streaming data; avoid the need for expensive data transformations, be robust to informal language, and avoid the determination and maintenance of a vocabulary as the language is constantly evolving. Next, we discuss the compression based distance, which addresses these requirements.

Tackling Data Informality

Compression distance computes the distance between two texts by measuring the compression gain obtained on the merging of the two texts. It has been shown to be both efficient and effective for clustering text in (Keogh, Lonardi, and Ratanamahatana 2004). Additionally, compression based distance has been shown to be effective on multilingual text. Although only discussed in the context of traditional documents, this distance measure is able to handle tweets due to its design. On Twitter, the advantage of compression distance over traditional distance measures such as cosine similarity, is its ability to handle the informal and evolving language in tweets. While cosine similarity requires the maintenance of a vocabulary and data transformation, compression distance can be directly applied to text.

Compression distance is an approximation of the Kolmogorov complexity proposed in (Keogh, Lonardi, and Ratanamahatana 2004). In this paper, we consider each tweet as a document. If C is any compressor, and $C(x)$ is the size of the compressed tweet x . Then the distance between two tweets x and y , $D(x, y)$ is

$$D(x, y) = \frac{C(xy)}{C(x) + C(y)}, \quad (4)$$

where $C(xy)$ is the compression achieved by merging the two tweets.

Using the above definition of compression distance between tweets, we can compute the distance between two events $D(e_1, e_2)$ as the maximum pairwise distance between any pair of tweets in the clusters.

Choosing the Compressor: Existing literature recommends choosing a compressor appropriate for the problem domain. In this paper, we compared 3 compression algorithms: DEFLATE, Gzip, and QuickLZ for compression speed and compression ratio using a random set of 20,000 tweets. We found that DEFLATE was the most efficient algorithm in both criteria. Therefore, we will employ it as the compressor **C** in Equation 4.

Scaling to High Volume Data

Twitter users currently generate more than 400 million tweets a day (Tsukayama 2013). Using publicly available Twitter APIs, one can access a sample of (1%) tweet stream, which can lead to as many as several million tweets a day. Thus, detecting events in a stream necessitates a scalable solution. Here, we present detailed solutions to scalability.

Events are dynamic and it is essential to consider the temporal evolution of the events in the task of event detection on streaming data. The incorporation of a temporal model into event detection has the following advantages:

- capturing evolving events, and
- improving the efficiency of event detection.

A cluster representing an event can be considered to be active or inactive at any given time based on the arrival of new tweets. Here, we propose a temporal model which can be used to make this decision. We model events as a Poisson process, which have been traditionally used to model the number of objects in an event at time t . In a Poisson process, the rate of arrival of tweets can be modeled as an exponential distribution. This rate is represented by the parameter λ . Let's consider a random variable X , where X measures the time between successive tweets. The variable X is modeled as an exponential distribution with parameter λ as

$$X \propto \exp(\lambda). \quad (5)$$

Given an event e and the number of tweets in each time interval in the event x_1, x_2, \dots, x_n , the likelihood function for the inter-arrival time is

$$L(\lambda|x_1, x_2, \dots, x_n) \propto f(x_1, x_2, \dots, x_n|\lambda) \propto \prod_{i=0}^n \lambda e^{-x_i \lambda}. \quad (6)$$

To obtain the λ_{MLE} , we take the derivative of the log-likelihood with respect to λ and set it to zero. Then, $\lambda_{MLE} = \frac{1}{\bar{x}}$, where \bar{x} is the mean of the distribution. For each cluster c , if a tweet does not arrive in λ_c time units, the current estimate for cluster c , then c is considered inactive and removed from memory. The estimate for λ_c is updated every time a new tweet is added to the cluster.

Identifying Events from Clusters

Tweets are noisy and not every tweet in the stream is expected to be part of an event. Therefore, not every cluster

ALGORITHM 1: Event Detection in Twitter Streams

Input: A stream of tweets T and the Cluster Limit (k), the Tweet Limit (l), the Distance Threshold (D_t), and the Diversity Threshold (H_t).

Output: Detected events E .

```

 $E \leftarrow \{\}$ ;
 $C \leftarrow \{\}$ ;
while tweet  $t \in T$  do
    Identify active cluster  $c \in C$ , where  $D(t, c) \leq D_t$ ;
    if  $c$  exists then
        Add  $t$  to  $c$ ;
        Update expected time of next tweet  $\lambda_c$ ;
        Update User Diversity ( $H(c)$ ) of cluster  $c$ ;
        if  $H(c) \geq H_t$  then
            Mark cluster as an event, Add  $c$  to  $E$ ;
        end
    end
    else
        Create new cluster  $c$  with  $t$  as its first member;
        Add  $c$  to  $C$ ;
    end
end

```

identified by the algorithm can be an event. The volume of a cluster can help us identify events, but this is susceptible to noise. As a crowdsourced information sharing platform, the diversity of the users (or the number of unique users) who publish tweets in a cluster lends credibility to the information within the cluster. Therefore, we measure the user diversity of a cluster to determine whether it is an event. A cluster is classified as an event, if its Diversity Score $H(c)$ is above the *Diversity Threshold* H_t .

Event Detection Framework

Using the strategies to handle informal language in tweets, temporal dynamics of events, and handling noise, we can detect events in Twitter streams using Algorithm 1. To improve the efficiency of the algorithm and to scale it to large Twitter streams we propose two heuristics:

Cluster Limit: The assignment of tweets to clusters requires a comparison with currently active clusters, but sequential search of all active clusters can be prohibitive. As a tweet is more likely to be similar to clusters with overlapping content we limit the comparisons to these candidate clusters. These clusters are identified by aggregating, sorting, and ranking clusters according to their overlap with the tweet. Then, we pick the top k clusters as the candidate clusters. $k = 100$, was empirically found to be effective in discovering reasonable clusters without sacrificing the speed of the algorithm.

Tweet Limit: The distance of a tweet to an event is computed as the maximum pairwise distance with the tweets contained in the event. Due to the timely nature of tweets, we propose to restrict the comparisons ordered by recency. This could be effective when clusters represent events which span an extended period of time and contain a large numbers of tweets. We propose to restrict the comparisons to at most l recent tweets in a cluster. In our implementation, we set $l = 1000$.

Time Complexity: Given the number of tweets in the stream

as N , the Cluster Limit (k) and the Tweet Limit (l), the time complexity of our algorithm is $O(Nkl)$. The most expensive operation in our algorithm is the assignment of tweets to clusters. For every tweet in the stream, it needs to be compared to at most l tweets in k clusters. As the values of k and l are much smaller than N , the algorithm allows us to process the tweets in near real-time. In the later sections, we will present empirical evidence of the algorithm’s efficiency.

Selection of Parameters: Two thresholds are used in our framework to identify events. First, the distance threshold D_t is used to determine assignment of tweets to clusters. In a study on 20,000 random tweets, we found that the average self-similarity of tweets was 0.54 and a value of 0.8 was empirically found to be a suitable value to obtain reasonable clusters. Second, the diversity threshold H_t is used to decide which clusters can be labeled as events, as noise is a problem in tweet streams. Volume or the number of tweets in a cluster is also an important factor in determining whether a cluster is an event. Ideally, we would like clusters to contain many tweets and have high user diversity. This threshold was set empirically as outlined later.

In the next section, we present evaluation results along: 1) scalability to high volume and high velocity streams, and 2) quality of the detected events.

Evaluation Strategy

There are two specific challenges in evaluating events from Twitter: 1) Unlike traditional media such as broadcast news, where every event is reported, on Twitter there is less likelihood of finding tweets related to minor events, and 2) While traditional research on event detection has relied upon the availability of labeled corpora such as the TDT corpus for evaluation, no such corpus exists for Twitter. Due to the lack of ground truth the exact number or nature of the events is not easily available and manual labeling of a large Twitter dataset is expensive. Twitter streams can be collected in two forms: **topic streams** containing tweets related to a specific topic, where the number and type of events can be verified using external sources, and **random streams**, which contain randomly sampled tweets, where the number and type of events must be manually determined. In this section, we evaluate the proposed approach on both types of streams.

Detecting Events in Topic Streams

As a representative topic stream, we introduce the Earthquake topic stream which consists of tweets related to earthquakes around the world.

Earthquakes: due to the existing research demonstrating the use of Twitter during earthquakes (Sakaki, Okazaki, and Matsuo 2010), (Mendoza, Poblete, and Castillo 2010), we collected tweets referring to earthquakes between June, 2011 to May, 2012 by monitoring the hashtags: #earthquake, #terremoto, and #quake. The data comprises of 1,007,417 tweets from 317,564 users.

To identify the real-world events spanned in this dataset, we must find an independent and external source, which can provide the ground truth at a reasonable cost as manual annotation is not practical. Towards this, we selected the major

Table 1: Major earthquakes in 2011 and 2012

Day(UTC)	Location	Magnitude	Death Toll
Jul 19, 2011	Fergana Valley	6.2	14
Sept 5, 2011	Aceh, Indonesia	6.7	12
Sept 18, 2011	India-Nepal border	6.9	111
Oct 23, 2011	Van, Turkey	7.1	684
Nov 9, 2011	Van, Turkey	5.7	40
Feb 6, 2012	Visayas, Philippines	6.7	113
Apr 11, 2012	Aceh, Indonesia	8.6	10
May 20, 2011	Emilia-Romagna, Italy	6.1 & 5.8	27

Table 2: Efficiency of event detection: Earthquake

Day	#tweets	Total processing time (Min)	Collection rate (Tweets/Min)	Processing rate (Tweets/Min)
Jul 19, 2011	880	0.04	0.613	23,498.00
Sept 5, 2011	2,712	0.18	1.88	14,788.69
Sept 18, 2011	465	0.02	0.32	18,699.73
Oct 23, 2011	5,253	0.49	3.65	10,646.89
Nov 9, 2011	2,712	0.17	1.89	15,611.63
Feb 6, 2012	13,586	4.79	13.72	2,834.92
Apr 11, 2012	28,182	10.61	19.57	2656.06
May 20, 2012	20,509	6.40	14.33	3,204.44

earthquakes in 2011 (Wikipedia 2011) and 2012 (Wikipedia 2012) on Wikipedia as the ground truth on the nature of the events in the dataset. These reports were manually compiled from several major news sources. In this paper, we focus our effort on the days when a major earthquake resulted in at least 10 casualties, which are summarized in Table 1. For most events in 2011, only a few hundred tweets were collected which might be due to the popularity of regional hashtags. Therefore, we set $H_t = 5$ for this dataset.

Evaluating Scalability To verify that our approach is scalable, we evaluated the rate at which the tweets in our dataset were generated and the time required by the proposed framework to identify events. Table 2 compares the measurements for the Earthquake dataset. Column 4 describes the rate at which tweets were collected and Column 5 describes the rate at which tweets were processed. We find that the proposed approach is capable of handling high volume topic-specific Twitter streams by being able to process the tweets at a rate which is significantly higher than the rate at which tweets were generated.

Quality of Detected Events Detected events are typically described using the frequent keywords from event tweets (Yang, Pierce, and Carbonell 1998; Petrovic, Osborne, and Lavrenko 2010; Fung et al. 2005). Therefore, we extracted the top keywords of each event as its description to verify whether they matched the ground truth. In Table 3, we present the most representative event corresponding to the events in Table 1. We also observed that the proposed approach was able to discover the evolution of events, which are represented by sub-events, which we will revisit later.

To quantify the effectiveness of our approach in detecting events, we compute the F_1 score which captures both

Table 3: Events detected in the Earthquake dataset

Day	Earthquake Location	Key Event Terms
Sept 5, 2011	Indonesia	sumatra, western, indonesian, island, #breakingnews
Oct 23, 2011	Turkey	#turkey, eastern, turkey, magnitude, news
Nov 9, 2011	Turkey	turkey, eastern, magnitude, rocks, usgs
Feb 6, 2012	Philippines	pray, visayas, philippines, struck, earlier
Apr 11, 2012	Indonesia	#indonesia, tsunami, magnitud, indonesia, sacudi
May 20, 2012	Italy	sentito, emilia, sono, cosa, chies

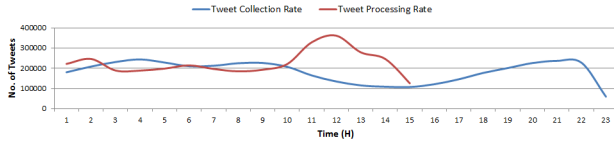


Figure 1: A comparison of the tweet collection rate and the tweet processing rate in the Random dataset

the Precision and Recall. Precision is computed as the number of detected events that match the ground truth including sub-events. Recall is computed as the number of events from the ground truth which were successfully detected. The F_1 Score for the Earthquake dataset was 0.77.

Detecting Events in a Random Stream

Twitter streams can also be collected without any topic bias using the Twitter Sample API¹. Using this API, we can retrieve a 1% random sample of the Twitter stream. The tweets in such streams include interpersonal conversations and discussions of real-world events. The task of event detection is harder in this case due to the presence of noise. To verify that our approach can be successfully applied to random streams, we collected sampled tweets from 11:02 AM on Apr 15 to 9:16 AM on Apr 16, 2013. The data consisted of 4,212,333 tweets from 3,322,379 users. As there is no ground truth for these days, we will test the effectiveness of our framework by verifying that we can detect the top stories of the day. We begin by establishing the scalability of our approach. Here we set $H_t = 6.3$ due to the larger volume.

Evaluating Scalability As in the case of the topic specific dataset, we test the efficiency of the proposed approach on a random stream by comparing the tweet generation rate and tweet processing rate. A comparison of these measurements is presented in Figure 1. The figure clearly shows that the processing speed for a majority of the collection period was on par with the collection speed and it often exceeded the tweet collection rate significantly. Nevertheless, the proposed approach was able to detect events in near-real time. This shows that our approach can be efficiently applied to a random Twitter stream.

Quality of Detected Events As manually labeling the tweets is not practical, we evaluate the quality of the events

Table 4: Events detected in the Random dataset

Event	Top Keywords
Venezuelan Presidential elections voting controversy	votos, capriles, esto, #caprilesgan-tibisayminti, fraude
Boston marathon bombing incident	marathon, boston, explosion, finish, line
Support for the bomb victims starts pouring in	boston, marathon, explosion, heart, bombing

based on the coverage of the two major events which occurred during this time period. From the random stream, we detected 167 events. A manual investigation of the events revealed 4 major kinds of events: events related to the Boston bombing incident, events related to the Presidential elections in Venezuela, events discussing a music festival, and finally events which represented banal Twitter chatter. An example of events expressing banal Twitter chatter included tweets from the fans of Justin Bieber, which resembled the characteristics of an event, but did not refer to a specific event. In Table 4, we present examples of the two main types of events: Boston marathon bombing and the Venezuelan Presidential elections. The first event discusses the controversy surrounding the counting of votes in the Presidential elections in Venezuela held on Apr 14, 2013 (Wikipedia 2013). The other two events are related to the Boston marathon bombing incident. The first event contains reports of the bombing itself and the second event references the reactions of the Twitter users. Our results show that we are able to detect reasonable events in the presence of large amount of noise.

Discussion

While few approaches exist to capture events in a streaming environment, the *Threading* technique proposed in (Petrovic, Osborne, and Lavrenko 2010) is the closest. Using the configuration recommended by the authors, we applied this technique to the Earthquake dataset. First, we compare the scalability of the two approaches. The results for this experiment are presented in Table 5. A comparison against our approach in Table 2 shows that our approach can process and detect events faster. Next, we evaluate the quality of the events. On all days, the Threading approach detected a greater number of events. Even using the ranking strategy proposed by the authors to retrieve the top 10 fastest growing events, we found that the F_1 score for the Threading technique was 0.64 compared to 0.77 for the proposed approach. As the Earthquake dataset was the smallest among our datasets, the results show that our framework outperforms this approach. The proposed approach also successfully removed noisy tweets.

An additional advantage of the proposed framework is that it can detect the evolution of events in dynamic Twitter streams. The inclusion of a *temporal model* allows us to identify sub-events within a larger event. For example, we can not only detect that an earthquake has occurred, but also detect the topics that emerge as a result of the earthquake, such as damage reports as seen in Table 6, where we present 5 events from the tweets generated during the Indonesian earthquake on April 11, 2012.

¹<https://dev.twitter.com/docs/api/1.1/get/statuses/sample>

Table 5: Efficiency of Threading technique: Earthquake

Day	#tweets	Total Processing Time (Min)	Tweet collection rate (Tweets/Min)	Processing rate (Tweets/Min)
Jul 19, 2011	880	1.11	0.613	793.40
Sept 5, 2011	2,712	3.99	1.88	678.68
Sept 18, 2011	465	0.88	0.32	527.10
Oct 23, 2011	5,253	2.65	3.65	1,984.97
Nov 9, 2011	2,712	2.54	1.89	1,068.13
Feb 6, 2012	13,586	38.36	13.72	354.19
Apr 11, 2012	28,182	135.27	19.57	208.34
May 20, 2012	20,509	210.32	14.33	97.51

Table 6: Evolution of events on April 11, 2012

Event	Top Keywords
Earthquake strikes Indonesia. Tsunami alert is issued	tsunami, #indonesia, #sumatra, scossa, allarme
Tremors felt in India	felt, singapore, thailand, indonesia, #tremors
Tsunami alert in Indian Ocean	tsunami, indian, ocean, move, alert
Sea water receding near the epicenter	aceh, quake, water, receding, island
Reports emerge that a tsunami is less likely	#indonesia, tsunami, moved, horizontally, vertically

Related Work

Event detection in traditional media is also known as Topic Detection and Tracking (TDT) and a pilot study on this task is presented in (Allan et al. 1998). In (Yang, Pierce, and Carbonell 1998), news articles were modeled as documents to detect topics. The documents were transformed into vector space using the TF-IDF and two clustering approaches were evaluated: Group-Average Agglomerative Clustering (GAAC) for retrospective event detection, and Incremental Clustering for new event detection. The authors concluded that the task of new event detection was harder. In (Allan, Papka, and Lavrenko 1998), the authors focused on on-line event detection. The authors approached the problem as a document-query matching problem. A query was constructed using the k most frequent words in a story. If a new document did not trigger existing queries then it was considered to be part of a new event. In (Fung et al. 2005), the authors addressed the problem of detecting *hot* bursty events. They introduced a new parameter-free clustering approach called feature-pivot clustering, which attempted to detect and cluster bursty features to detect hot stories.

An attempt to detect earthquakes using Twitter users as social sensors was carried out by in (Sakaki, Okazaki, and Matsuo 2010). The temporal aspect of an event was modeled as an exponential distribution, and the probability of the event was determined based on the likelihood of each sensor being incorrect. (Becker, Naaman, and Gravano 2010) tackled event detection in Flickr. The authors leveraged the meta data of images to create both textual and non-textual features and proposed the use of individual distance measures for each feature. These features were used to create independent partitions of the data and finally the partitions were combined using a weighted ensemble scheme to detect event clusters. In (Weng and Lee 2011), the authors con-

structed word signals using the Wavelet Transformation and used a modularity-based graph partitioning approach on the correlation matrix to get event clusters. (Li, Sun, and Datta 2012) identified bursty segments in tweets and clustered the segments to identify events.

Few existing approaches are designed for streaming Twitter data and even fewer are scalable to real-time streams. In (Sayyadi, Hurst, and Maykov 2009), the authors converted a stream of blog posts into a keyword graph, where nodes represented words and links represented co-occurrence. Community detection methods were applied on the graph to detect communities of related words or events. In (Zhao, Mitra, and Chen 2007), the authors model the social text streams including blogs and emails as a multi-graph and cluster the streams using textual, temporal, and social information to detect events. A hybrid network and content based clustering approach was employed in (Aggarwal and Subbian 2012) to identify a fixed number of events in a labeled Twitter stream containing tweets from two events. Generally, the number of events is not known beforehand and obtaining the user network adds significant overhead, thus adding to the complexity of this method. In (Petrovic, Osborne, and Lavrenko 2010), the authors recognized the need for faster approaches for first story detection in streams. The authors proposed a two-step process to identify first stories in streaming data. First, the nearest neighbor of each tweet is identified using locally sensitive hashing in constant time and space. Second, a clustering approach called Threading is applied to group related tweets into event clusters. The first tweet in a thread is presented as the first story and the thread itself is considered an event.

Conclusions and Future Work

In this work, we presented a novel approach to detect events in informal and high volume Twitter streams. The results demonstrate that the proposed approach can handle the informality of language in Twitter streams, through the use of compression distance. We found that the proposed approach is capable of handling dynamic streams, where the number of events is unknown or cannot be practically determined in near real-time. The proposed User Diversity measure is also able to successfully filter noise in Twitter streams. Through experiments we demonstrated that the proposed approach is efficient and is able to capture reasonable events in topic streams and random streams on Twitter.

Event detection has several potential applications, which we intend to investigate as part of our future work. Investigating the relationship between an event's rate of growth and its impact in the real-world is one. Another direction of future study is the categorization of events based on two characteristics: the volume of the event defined by the number of tweets, and the rate of the event. By organizing the events in this fashion, we can provide a value added service to a user by facilitating the tracking of specific types of events.

Acknowledgments

This work was sponsored, in part, by the Office of Naval Research grant N000141410095.

References

- [Aggarwal and Subbian 2012] Aggarwal, C. C., and Subbian, K. 2012. Event Detection in Social Streams. In *SDM*, 624–635.
- [Allan et al. 1998] Allan, J.; Carbonell, J.; Doddington, G.; Yamron, J.; and Yang, Y. 1998. Topic Detection and Tracking Pilot Study Final Report.
- [Allan, Papka, and Lavrenko 1998] Allan, J.; Papka, R.; and Lavrenko, V. 1998. On-Line New Event Detection and Tracking. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, 37–45. ACM.
- [Analytics 2009] Analytics, P. 2009. Twitter Study. <http://www.pearanalytics.com/wp-content/uploads/2012/12/Twitter-Study-August-2009.pdf>. [Online; accessed 27-January-2014].
- [Becker, Naaman, and Gravano 2010] Becker, H.; Naaman, M.; and Gravano, L. 2010. Learning Similarity Metrics for Event Identification in Social Media. In *Proceedings of the third ACM international conference on Web search and data mining*, 291–300. ACM.
- [Fung et al. 2005] Fung, G. P. C.; Yu, J. X.; Yu, P. S.; and Lu, H. 2005. Parameter Free Bursty Events Detection in Text Streams. In *Proceedings of the 31st international conference on Very large data bases*, 181–192. VLDB Endowment.
- [Gilgoff and Lee 2013] Gilgoff, D., and Lee, J. J. 2013. Social Media Shapes Boston Bombings Response. <http://bit.ly/1iEExb>. [Online; accessed 27-January-2014].
- [Huang 2011] Huang, C. 2011. Facebook and Twitter key to Arab Spring uprisings: report. <http://bit.ly/1bh6jV6>. [Online; accessed 28-August-2013].
- [Keogh, Lonardi, and Ratanamahatana 2004] Keogh, E.; Lonardi, S.; and Ratanamahatana, C. 2004. Towards Parameter-Free Data Mining. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 206–215. ACM.
- [Khamadi Were 2013] Khamadi Were, D. 2013. How Kenya turned to social media after mall attack. http://edition.cnn.com/2013/09/25/opinion/kenya-social-media-attack/index.html?hpt=hp_c4. [Online; accessed 27-January-2014].
- [Krause and Golovin 2012] Krause, A., and Golovin, D. 2012. Submodular function maximization. *Tractability Practical Approaches to Hard Problems* 3.
- [Li, Sun, and Datta 2012] Li, C.; Sun, A.; and Datta, A. 2012. Twevent: Segment-based event detection from tweets. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, 155–164. ACM.
- [Mendoza, Poblete, and Castillo 2010] Mendoza, M.; Poblete, B.; and Castillo, C. 2010. Twitter Under Crisis: Can we Trust What We RT? In *Proceedings of the First Workshop on Social Media Analytics*.
- [Paris, Thomas, and Wan 2012] Paris, C.; Thomas, P.; and Wan, S. 2012. Differences in language and style between two social media communities. In *Proceedings of 6th AAAI International Conference on Weblogs and Social Media*.
- [Petrovic, Osborne, and Lavrenko 2010] Petrovic, S.; Osborne, M.; and Lavrenko, V. 2010. Streaming First Story Detection with Application to Twitter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, volume 10. Citeseer.
- [Qu et al. 2011] Qu, Y.; , C.; Zhang, P.; and Zhang, J. 2011. Microblogging After a Major Disaster in China: A Case Study of the 2010 Yushu Earthquake. In *Proceedings of the ACM 2011 conference on Computer supported cooperative work*, 25–34.
- [Rijsbergen 1979] Rijsbergen, C. J. V. 1979. *Information Retrieval*. Newton, MA, USA: Butterworth-Heinemann, 2nd edition.
- [Sakaki, Okazaki, and Matsuo 2010] Sakaki, T.; Okazaki, M.; and Matsuo, Y. 2010. Earthquake Shakes Twitter Users: Real-Time Event Detection by Social Sensors. In *Proceedings of the 19th international conference on World wide web*, 851–860. ACM.
- [Sayyadi, Hurst, and Maykov 2009] Sayyadi, H.; Hurst, M.; and Maykov, A. 2009. Event Detection and Tracking in Social Streams. In *Proceedings of 3rd AAAI International Conference on Weblogs and Social Media*.
- [Tsukayama 2013] Tsukayama, H. 2013. Twitter turns 7: Users send over 400 million tweets per day. http://articles.washingtonpost.com/2013-03-21/business/37889387_1_tweets-jack-dorsey-twitter. [Online; accessed 25-September-2013].
- [Weng and Lee 2011] Weng, J., and Lee, B. S. 2011. Event detection in twitter. In *ICWSM*.
- [Wikipedia 2011] Wikipedia. 2011. Earthquakes in 2011. http://en.wikipedia.org/wiki/Earthquakes_in_2011. [Online; accessed 27-January-2014].
- [Wikipedia 2012] Wikipedia. 2012. Earthquakes in 2012. http://en.wikipedia.org/wiki/Earthquakes_in_2012. [Online; accessed 27-January-2014].
- [Wikipedia 2013] Wikipedia. 2013. Venezuelan presidential election, 2013. http://en.wikipedia.org/wiki/Venezuelan_presidential_election,_2013. [Online; accessed 27-January-2014].
- [Yang, Pierce, and Carbonell 1998] Yang, Y.; Pierce, T.; and Carbonell, J. 1998. A Study of Retrospective and On-Line Event Detection. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, 28–36. ACM.
- [Zhao, Mitra, and Chen 2007] Zhao, Q.; Mitra, P.; and Chen, B. 2007. Temporal and Information Flow Based Event Detection From Social Text Streams. In *AAAI*, volume 7, 1501–1506.